PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L 12/56 — A1

(11) International Publication Number: WO 99/00950

(43) International Publication Date: 7 January 1999 (07.01.99)

(21) International Application Number: PCT/US98/13368

(22) International Filing Date: 25 June 1998 (25.06.98)

(30) Priority Data:
08/885,233 30 June 1997 (30.06.97) US

(71) Applicant: SUN MICROSYSTEMS, INC. [US/US]; 901 San Antonio Road, Palo Alto, CA 94303 (US).

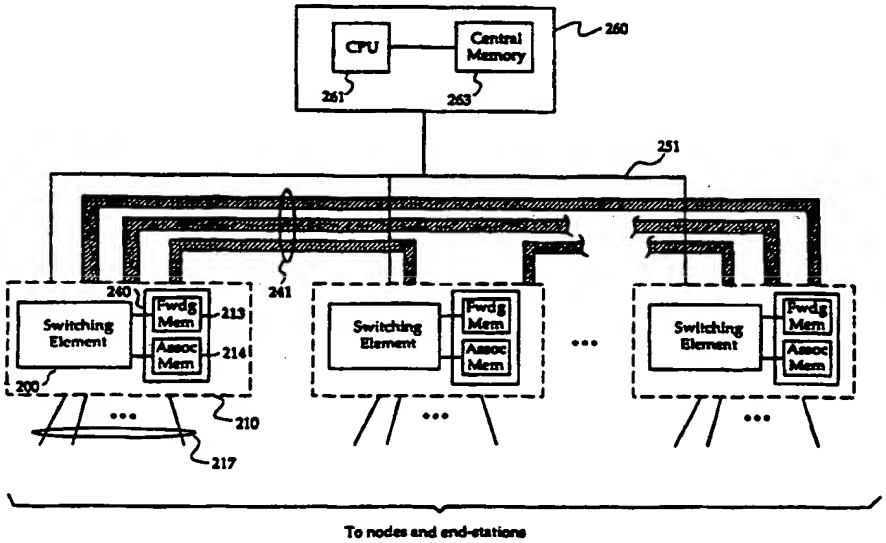
(72) Inventors: MULLER, Shimon; Apartment D, 983 La Mesa Terrace, Sunnyvale, CA 94086 (US). HENDEL, Ariel; 7537 Newcastle Drive, Cupertino, CA 95014 (US).

(74) Agents: HYMAN, Eric, S. et al.; Blakely, Sokoloff, Taylor & Zafman, 7th floor, 12400 Wilshire Boulevard, Los Angeles, CA 90025-1026 (US).

(81) Designated States: JP, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.

(54) Title: TRUNKING SUPPORT IN A HIGH PERFORMANCE NETWORK DEVICE



(57) Abstract

A method and apparatus for providing trunking support in a network device (201) is provided. The network device includes at least one port that is configured to be included in a trunk and a memory for storing a forwarding database (240). The forwarding database includes entries containing therein forwarding information for a subset of network addresses. The network device further includes a learning circuit (260) coupled to the trunked port and the memory. The learning circuit is configured to modify the forwarding database to reflect an association between the trunked port and a first address contained within a packet received by the trunked port. If the trunk is of a first type, the learning circuit updates the forwarding database based upon a trunk designator corresponding to the trunk, otherwise the learning circuit updates the forwarding data base based upon a port designator corresponding to the trunked port.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

TRUNKING SUPPORT IN A HIGH PERFORMANCE NETWORK DEVICE

FIELD OF THE INVENTION

The invention relates generally to the field of computer networking devices. More particularly, the invention relates to a network device building block that facilitates combining multiple parallel physical network links into one logical channel.

BACKGROUND OF THE INVENTION

Generally, trunking can be thought of as a means of providing bandwidth aggregation between two points in a network (e.g., between two network devices). Figure 1 is useful for illustrating the concept of trunking. A first device 105 and a second device 110 are connected through a plurality of physical network links 115-117. The first device 105 and the second device 110 may be network devices, such as a server, client, repeater, bridge, router, brouter, switch, or the like. The first device 105 includes ports 106-109 and the second device 110 includes ports 111-114. The ports provide the device with access to the attached network link by implementing appropriate network protocols such as the Ethernet protocol.

In this example, the physical network links 115-117 have been combined to form one logical channel, a "trunk" 140, between the first device 105 and the second device 110. As mentioned above, a trunk may provide increased bandwidth between two points in a network. For example, if links 115-117 each individually have a bandwidth of 100 Mbps, the resulting bandwidth of the trunk 140 is the sum of the bandwidths of the individual links (100 Mbps + 100 Mbps + 100 Mbps = 300 Mbps).

At this point, it is important to recognize that two types of network devices have emerged. The first type of device (hereinafter "MODE 1 device") has the same media access control (MAC) address on its trunked ports. The second type of device (hereinafter "MODE 2 device") has a different MAC address on each trunked port.

One limitation of conventional switches is the fact that they are unable to perform upstream load balancing for MODE 1 devices. For example, assuming ports 106-108 of the first device 105 each use the MAC address of the first device 105, the second device 110 will relearn the location for that MAC address each time the first device 105 transmits a packet over a different trunked port 106-108. Consequently, packet traffic destined for the first device 105 over trunk 140 cannot be distributed over the ports 111-113. Rather, the port on which the second device 110 will transmit these packets depends upon which of the ports 106-108 transmitted last.

Based on the foregoing, it is desirable to implement a set of trunking rules relating to learning, forwarding, looping, and load balancing that are compatible with network devices operating in either MODE 1 or MODE 2. Also, in order to avoid introducing delays in packet transmission, it is desirable to update packet forwarding decisions that will be affected by trunk processing on-the-fly.

SUMMARY OF THE INVENTION

A method and apparatus for providing trunking support in a network device is described. According to one aspect of the present invention, a network device includes at least one port that is configured to be included in a trunk. The network device also includes a memory for storing a forwarding database. The forwarding database includes entries containing therein forwarding information for a subset of network addresses. The network device further includes a learning circuit. The learning circuit is coupled to the trunked port and the memory. The learning circuit is configured to modify the forwarding database to reflect an association between the trunked port and a first address contained within a packet received by the trunked port. If the trunk is of a first type, the learning circuit updates the forwarding database based upon a trunk designator corresponding to the trunk; otherwise, the learning circuit updates the forwarding database based upon a port designator corresponding to the trunked port.

According to another aspect of the present invention, a network device includes at least one port that is included in a trunk associated with a second network device. The network device also includes a memory for storing a forwarding database. The forwarding database includes entries that each contain forwarding information for a particular network address. The network device further includes a filtering circuit coupled to the trunked port and the memory for receiving forwarding information corresponding to a destination address contained within a packet. Based upon a predetermined set of trunking rules and one or more characteristics of the trunk, the filtering circuit is configured to modify the forwarding information generated by the forwarding database.

According to another aspect of the present invention, a packet is received by a network device on one of its input ports. The packet contains therein header information including the packet's destination address. A forwarding database is searched for the destination address. If the destination address is not found in the forwarding database and the output trunk is coupled to a network device of a first type, then load balancing is performed to assure the packet is forwarded to only one port of the output trunk. However,

if the output trunk is coupled to a network device of a second type, then the packet is forwarded to all ports of the output trunk.

Other features of the present invention will be apparent from the accompanying drawings and from the detailed description which follows.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

Figure 1 illustrates two devices coupled in communication via a trunk.

Figure 2 illustrates a switch according to one embodiment of the present invention.

Figure 3 is a high level block diagram of a switch element in which one embodiment of the present invention may be implemented.

Figure 4 is a block diagram which illustrates the interaction of trunk forwarding circuitry and trunk learning circuitry according to one embodiment of the present invention.

Figure 5 is a high level flow diagram illustrating trunk processing according to one embodiment of the present invention.

Figure 6A is a block diagram which illustrates an exemplary trunk learning module according to one embodiment of the present invention.

Figure 6B is a flow diagram illustrating Layer 2 learning according to one embodiment of the present invention.

Figure 6C illustrates exemplary logic for implementing the trunk learning module of Figure 6A.

Figure 7A is a block diagram which illustrates an exemplary trunk filtering module according to one embodiment of the present invention.

Figure 7B is a flow diagram illustrating trunk filtering according to one embodiment of the present invention.

Figure 7C is a flow diagram illustrating load balancing according to one embodiment of the present invention.

Figure 7D illustrates exemplary logic for implementing the trunk filtering module of Figure 7A.

DETAILED DESCRIPTION

A method and apparatus are described for providing trunking support in a network device. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without some of these specific details. In other instances, well-known structures and devices are shown in block diagram form.

The present invention includes various steps, which will be described below. While the steps of the present invention are preferably performed by the hardware components described below, alternatively, the steps may be embodied in machine-executable instructions, which may be used to cause a general-purpose or special-purpose processor programmed with the instructions to perform the steps.

AN EXEMPLARY NETWORK ELEMENT

An overview of one embodiment of a network element that operates in accordance with the teachings of the present invention is illustrated in Figure 2. The network element is used to interconnect a number of nodes and end-stations in a variety of different ways. In particular, an application of the multi-layer distributed network element (MLDNE) would be to route packets according to predefined routing protocols over a homogenous data link layer such as the IEEE 802.3 standard, also known as the Ethernet. Other routing protocols can also be used.

The MLDNE's distributed architecture can be configured to route message traffic in accordance with a number of known or future routing algorithms. In a preferred embodiment, the MLDNE is configured to handle message traffic using the Internet suite of protocols, and more specifically the Transmission Control Protocol (TCP) and the Internet Protocol (IP) over the Ethernet LAN standard and medium access control (MAC) data link layer. The TCP is also referred to here as a Layer 4 protocol, while the IP is referred to repeatedly as a Layer 3 protocol.

In one embodiment of the MLDNE, a network element is configured to implement packet routing functions in a distributed manner, i.e., different parts of a function are performed by different subsystems in the MLDNE, while the final result of the functions remains transparent to the external nodes and end-stations. As will be appreciated from the discussion below and the diagram in Figure 2, the MLDNE has a scalable architecture which allows the designer to predictably increase the number of external connections by adding

additional subsystems, thereby allowing greater flexibility in defining the MLDNE as a stand alone router.

As illustrated in block diagram form in Figure 2, the MLDNE 201 contains a number of subsystems 210 that are fully meshed and interconnected using a number of internal links 241 to create a larger switch. At least one internal link couples any two subsystems. Each subsystem 210 includes a switch element 200 coupled to a forwarding and filtering database 240, also referred to as a forwarding database. The forwarding and filtering database may include a forwarding memory 213 and an associated memory 214. The forwarding memory (or database) 213 stores an address table used for matching with the headers of received packets. The associated memory (or database) stores data associated with each entry in the forwarding memory that is used to identify forwarding attributes for forwarding the packets through the MLDNE. A number of external ports (not shown) having input and output capability interface the external connections 217. In one embodiment, each subsystem supports multiple Gigabit Ethernet ports, Fast Ethernet ports and Ethernet ports. Internal ports (not shown) also having input and output capability in each subsystem couple the internal links 241. Using the internal links, the MLDNE can connect multiple switching elements together to form a multigigabit switch.

The MLDNE 201 further includes a central processing system (CPS) 260 that is coupled to the individual subsystem 210 through a communication bus 251 such as the peripheral components interconnect (PCI). The CPS 260 includes a central processing unit (CPU) 261 coupled to a central memory 263. Central memory 263 includes a copy of the entries contained in the individual forwarding memories 213 of the various subsystems. The CPS has a direct control and communication interface to each subsystem 210 and provides some centralized communication and control between switch elements.

AN EXEMPLARY SWITCH ELEMENT

Figure 3 is a simplified block diagram illustrating an exemplary architecture of the switch element of Figure 2. The switch element 200 depicted includes a central processing unit (CPU) interface 315, a switch fabric block 310, a network interface 305, a cascading interface 325, and a shared memory manager 320.

Ethernet packets may enter or leave the network switch element 200 through any one of the three interfaces 305, 315, or 325. In brief, the network interface 305 operates in accordance with a corresponding Ethernet protocol to receive Ethernet packets from a network (not shown) and to transmit Ethernet packets onto the network via one or more

external ports (not shown). An optional cascading interface 325 may include one or more internal links (not shown) for interconnecting switching elements to create larger switches. For example, each switch element may be connected together with other switch elements in a full mesh topology to form a multi-layer switch as described above. Alternatively, a switch may comprise a single switch element 200 with or without the cascading interface 325.

The CPU 261 may transmit commands or packets to the network switch element 200 via the CPU interface 315. In this manner, one or more software processes running on the CPU may manage entries in an external forwarding and filtering database 240, such as adding new entries and invalidating unwanted entries. In alternative embodiments, however, the CPU may be provided with direct access to the forwarding and filtering database 240. In any event, for purposes of packet forwarding, the CPU port of the CPU interface 315 resembles a generic input port into the switch element 200 and may be treated as if it were simply another external network interface port. However, since access to the CPU port occurs over a bus such as a peripheral components interconnect (PCI) bus, the CPU port does not need any media access control (MAC) functionality.

Returning to the network interface 305, the two main tasks of input packet processing and output packet processing will now briefly be described. Input packet processing may be performed by one or more input ports of the network interface 305. Input packet processing includes the following: (1) receiving and verifying incoming Ethernet packets, (2) modifying packet headers when appropriate, (3) requesting buffer pointers from the shared memory manager 320 for storage of incoming packets, (4) requesting forwarding decisions from the switch fabric block 310, (5) transferring the incoming packet data to the shared memory manager 320 for temporary storage in an external shared memory 230, and (5) upon receipt of a forwarding decision, forwarding the buffer pointer(s) to the output port(s) indicated by the forwarding decision. Output packet processing may be performed by one or more output ports of the network interface 305. Output processing includes requesting packet data from the shared memory manager 320, transmitting packets onto the network, and requesting deallocation of buffer(s) after packets have been transmitted.

The network interface 305, the CPU interface 315, and the cascading interface 325 are coupled to the shared memory manager 320 and the switch fabric block 310. Preferably, critical functions such as packet forwarding and packet buffering are centralized as shown in Figure 3. The shared memory manager 320 provides an efficient centralized interface to the external shared memory 230 for buffering of incoming packets. The switch fabric block 310

includes a search engine and learning logic for searching and maintaining the forwarding and filtering database 240 with the assistance of the CPU.

The centralized switch fabric block 310 includes a search engine that provides access to the forwarding and filtering database 240 on behalf of the interfaces 305, 315, and 325. Packet header matching, Layer 2 based learning, Layer 2 and Layer 3 packet forwarding, filtering, and aging are exemplary functions that may be performed by the switch fabric block 310. Each input port is coupled with the switch fabric block 310 to receive forwarding decisions for received packets. The forwarding decision indicates the outbound port(s) (e.g., external network port or internal cascading port) upon which the corresponding packet should be transmitted. Additional information may also be included in the forwarding decision to support hardware routing such as a new MAC destination address (DA) for MAC DA replacement. Further, a priority indication may also be included in the forwarding decision to facilitate prioritization of packet traffic through the switch element 200.

In the present embodiment, Ethernet packets are centrally buffered and managed by the shared memory manager 320. The shared memory manager 320 interfaces every input port and output port and performs dynamic memory allocation and deallocation on their behalf, respectively. During input packet processing, one or more buffers are allocated in the external shared memory 230 and an incoming packet is stored by the shared memory manager 320 responsive to commands received from the network interface 305, for example. Subsequently, during output packet processing, the shared memory manager 320 retrieves the packet from the external shared memory 230 and deallocates buffers that are no longer in use. To assure no buffers are released until all output ports have completed transmission of the data stored therein, the shared memory manager 320 preferably also tracks buffer ownership.

The present invention may be included in a switch element such as switch element 200. However, the method and apparatus described herein are equally applicable to other types of network devices such as repeaters, bridges, routers, brouters, and other network devices.

OVERVIEW OF TRUNKING RULES AND CONCEPTS

Load balancing, e.g., the spreading of packet traffic over different links of a trunk, and avoidance of packet duplication are objectives that should be satisfied by the two network devices connected to both ends of the trunk. The present invention employs the following concepts and adheres to the following rules in order to achieve these goals.

Both MODE 1 and MODE 2 devices are supported and mixing and matching among ports is allowed. One or more programmable registers or other memory may store an indication for differentiating between the two modes. When the indication is in a first state, learning and forwarding processing for the particular port are configured for compatibility with MODE 1 devices, for example. Similarly, when the indication is in a second state, learning and forwarding processing are configured for compatibility with MODE 2 devices.

Before discussing the rules and concepts related to learning, Layer 2 based learning will briefly be described. Layer 2 based learning is the process of constantly updating the MAC address portion of the forwarding database based on the traffic that passes through the switching device. When a packet enters the switching device, an entry is created (or an existing entry is updated) in the database that correlates the MAC source address of the packet with the input port upon which the packet arrived. Multicast addresses are set up by software rather than learned by the hardware. In any event, between the learning process and software mapping of destination addresses to ports, the switching device knows on which subnet a given node resides.

Continuing now with the rules and concepts related to learning, depending on the type of device that originated the packet (e.g., MODE 1 or MODE 2 device), trunk numbers or port numbers may be learned and/or stored in the forwarding and filtering database. As described in the background, packets originating from a MODE 1 device will have the same MAC SA in their headers regardless of the port upon which they are transmitted. Consequently, if a port number were to be associated with the MAC SA, the MAC SA would be relearned upon receipt of each subsequent packet transmitted by the device on a different port. Therefore, for trunked ports coupled to MODE 1 devices, it is better to use a trunk number for purposes of learning. In contrast, packets received from a MODE 2 device will have a different MAC SA depending on the port upon which the packet was transmitted. For these types of devices, it is convenient to associate a port number with the MAC SA. At any rate, the learned port or trunk number may be encoded as a "port mask". For example, a set of N bits may be used to encode a port forwarding mask for N ports. When the bit in position X of the set of N bits is in a first state, the packet is to be forwarded to port X. However, when the bit is in a second state, the packet is to be filtered. Of course, those of ordinary skill in the art will appreciate that alternative representations may be used.

Throughout this application it will be assumed masks are employed to represent the portion of a forwarding decision indicating outbound ports. With respect to forwarding, therefore, a unicast packet will only have one bit set to the forwarding state in the port mask provided to the input port that requested the forwarding decision. However, the port mask

corresponding to a multicast packet may have several bit positions set to the forwarding state. Also, unknown unicast packets (e.g., those packets having destination address that have not been learned by this network device) may have all bits set to the forwarding state (e.g., to flood the packet).

At this point it may be instructive to explain the usage of the terms "known" and "unknown" in the context of a learning and forwarding network device. Packets are said to be known when the packet's destination address has been learned and is found in the forwarding database. In contrast, packets are said to be unknown when the packet's destination address has not been learned or is not found in the forwarding database.

With respect to forwarding, any packet (known unicast, multicast, or unknown unicast) should not be received on more than one port of a multi-homed device. Since all ports of a MODE 1 device have the same MAC address, all ports of the MODE 1 device will respond in the same manner to a received packet. That is, all ports will either accept the packet or all ports will filter the packet. Therefore, it is important to assure that only one packet will be forwarded to trunked ports coupled to MODE 1 devices. In contrast, since the ports of MODE 2 devices each have their own MAC addresses, only one port will be capable of receiving a given unicast packet. Given these characteristics of MODE 1 and MODE 2 devices, load balancing is applied to all packets forwarded to MODE 1 devices and load balancing only needs to be applied to multicast packets forwarded to MODE 2 devices. Further, unknown unicast packets are forwarded to only one port of a trunk coupled to a MODE 1 device and may be forwarded to all ports of a trunk coupled to a MODE 2 device.

With respect to looping, any packet (known unicast, multicast or unknown unicast) arriving on a trunk should not be forwarded to other ports of the same trunk with the exception of a packet that is part of a Layer 3 protocol unicast route.

Several simplifying assumptions may also be made to reduce the complexity of trunk processing. For example, trunking may always be considered enabled. Given this assumption, a single port may be treated as a trunk of size one. Also, if the trunked ports are contiguously assigned and the trunk number is defined as the smallest port number in a trunk other advantages can be achieved as will be understood from the discussion below. Further, load balancing may be approximated. The packet traffic on a particular trunk need not be shared precisely among the participating ports. All that is needed is a mechanism to somewhat randomize the flow of packets through the trunk to assure that a particular trunked port is not over or under utilized. This balancing can be achieved in a number of ways. One method is to employ a hash function or the like as will be described in more detail below.

TRUNK LEARNING AND FILTERING

Having briefly described the rules and concepts related to load balancing, learning, forwarding, and looping, exemplary logic and steps for implementing these rules will now be described. Figure 4 is a block diagram of trunk logic within the switch fabric 310 according to one embodiment of the present invention. In this embodiment, a trunk register is provided for each of N ports. Port 1 corresponds to a first trunk register 410, Port N corresponds to the last trunk register 420. Preferably, each register is of the form of registers 410 and 420. Register 410 includes a trunk number field 411, a use trunk number field 412, and a trunk size field 413. The trunk number field 411, in this example, is a four bit value that stores a trunk number corresponding to the trunk in which this port is a member. In one embodiment, the trunk number is the lowest port number of the ports in the trunk. This choice of trunk number has advantages that will be discussed below with respect to load balancing.

Continuing with this example, a one bit field, the use trunk number field 412, is used to indicate whether or not to use the trunk number 411 for the particular port for purposes of executing learning and forwarding. Typically, the port number is used for ports attached to MODE 2 devices and the trunk number is used for ports attached to MODE 1 devices. Finally, in the embodiment depicted, the trunk size 413 is a three bit field for indicating the number of ports in the trunk. While, for purposes of this example, the trunk characteristics have been described as being stored in registers, it will be recognized that numerous other storage mechanisms are possible.

The trunk learning and filtering logic of the present invention also includes a trunk filter block 430 for each port and a common trunk learning block 440. The registers are coupled to the corresponding filter block 430 and the learning block 440 to provide the trunk information to these blocks. Other inputs to the blocks may include a portion of the most significant bits (MSBs) of the packet's destination address, the individual port mask bits from the filtering and forwarding database, the input trunk number, the input port number, a portion of the least significant bits (LSBs) of the packet's source address, and Layer 3 information such as whether or not the packet is part of a unicast route. Given the inputs described above, during the learning process, the learning block 440 produces a port mask (e.g., LearnPortMask[N:1]) that will be stored in the forwarding and filtering database. Further, during the forwarding process, each filter block 430 contributes a bit toward a final forwarding port mask (e.g., FwdPortMask[N:1]). The final forwarding port mask is ultimately communicated to the input port that requested the forwarding decision for this particular packet.

TRUNK PROCESSING OVERVIEW

Having described an exemplary environment in which the present invention may be implemented, trunk processing will now be described. Figure 5 is a high level flow diagram illustrating trunk processing according to one embodiment of the present invention. At step 510, a packet is received on an input port of the network interface 305 or the cascading interface 325. At the appropriate point during reception of the incoming packet, the input port requests from the switch fabric block 310 a forwarding decision and a learn cycle.

At step 520, learning is performed. Conventionally, learning is the association of an incoming packet's MAC SA and its port of arrival. However, the present invention accommodates trunking by allowing a representation of a trunk designator to be associated with the incoming packet's MAC SA in certain circumstances which will be described below with respect to Figure 6B. In any event, during the learning process, the forwarding and filtering database is updated to reflect the new information learned as a result of the newly received packet. For example, if the MAC SA is not found in the forwarding and filtering database, then a new entry is created and stored in the forwarding and filtering database so subsequent packets destined for that MAC address can be forwarded appropriately. Alternatively, if the MAC SA is found the port or trunk associated with the MAC address is updated and an age indication is cleared.

At step 530, trunk filtering is performed. Trunk filtering is the mechanism by which one or more ports of a particular trunk are selected on which to forward the current packet. The filtering of step 530 is performed on the fly upon data such as a port list that has been retrieved from the forwarding and filtering database. In this example, it is assumed that the port list (e.g., the portion of the forwarding decision that indicates the ports on which to forward the packet) is a "port mask". It should be appreciated that if the port mask retrieved from the forwarding and filtering database indicates a packet should be filtered for a particular port, this decision will not be reversed by the trunk filtering logic. As the name implies, the trunk filtering logic will only change a bit in the port mask from the forward state to the filter state.

At step 540, after all output ports have performed the filtering logic of step 530, processing continues with step 550. While the filtering logic of step 530 may be performed serially as depicted in the flow diagram, it is appreciated multiple circuits may be used to perform the filtering for each port in parallel. Finally, at step 550, the forwarding decision is returned to the input port from which it was requested.

EXEMPLARY TRUNK LEARNING MODULE

Figure 6A is a block diagram which illustrates an exemplary trunk learning module according to one embodiment of the present invention. In this embodiment, the trunk learning block 440 includes input port trunk characteristics selection logic 611, input port number/trunk number selection logic 612, and a mask generator 614. The input port trunk characteristics selection logic 611 receives the trunk characteristics of each trunk and outputs the characteristics corresponding to the trunk to which the input port belongs.

The input port number/trunk number selection logic 612 is coupled to the trunk characteristics selection logic 611 to receive the usebit and the trunk number for the input port. The input port number/trunk number selection logic 612 outputs the trunk number if the usebit is in a first state that indicates the trunk number is to be used, otherwise the input port number is output.

The mask generator 614 is coupled to the output of the input port number/trunk number selection logic 612 for receiving the input trunk number or the input port number. According to this embodiment, a port mask (e.g., LearnPortMask[N:1]) is generated that corresponds to the number input from the input port number/trunk number selection logic 612. For example, if the mask generator 614 receives the number 5 and the network device has N=8 ports, then a port mask of 00010000 might be generated. The 5th bit of the eight bit mask is set indicating packets destined for the address being learned should be forwarded to port or trunk number 5 depending upon the state of the usebit.

LEARNING PROCESSING

Figure 6B is a flow diagram illustrating a method of Layer 2 learning according to one embodiment of the present invention. At step 613, a set of data representing the trunk information corresponding to the input port is selected.

At step 622, a determination is made as to whether the trunk number should be used or the port number. In this embodiment, the determination is made with reference to the type of device (e.g., MODE 1 or MODE 2) from which the packet was received. If the port number is to be used, the learning processing continues with step 623, otherwise the processing continues with step 625.

At step 623, a representation of the port number such as a port mask is generated. At step 624, the port mask is stored in the forwarding and filtering database.

If the trunk number is to be used, a representation of the trunk number, in the form of a mask, for example, is generated in step 625. Then, at step 626, the trunk mask is stored in the forwarding and filtering database at step.

EXEMPLARY TRUNK LEARNING LOGIC

Figure 6C illustrates exemplary logic for implementing the trunk learning module of Figure 6A. In this embodiment, the trunk characteristics selection logic 611 includes a multiplexer (MUX) 610. The trunk number output from the MUX 610 and the input port number are inputs to a MUX 615. The usebit output of MUX 610 is the select input to MUX 615. The output of MUX 615 is coupled to the input of a decoder 620, which implements the port mask generation.

With respect to the portion of the load balance logic 640 shown in the trunk leaning block 440, the exclusive or logic 644 produces a hash result based upon the input port number and a portion of the source address of the incoming packet. Other methods of generating a pseudo random or random number such as a random number generator may alternatively be employed. In any event, so long as the logic that produces the LoadBalanceHash is not dependent upon the output port it is preferable to avoid duplicating this common function unnecessarily. However, this logic may be located in the trunk filtering block 430 or elsewhere in alternative embodiments.

EXEMPLARY TRUNK FILTERING MODULE

Figure 7A is a block diagram which illustrates an exemplary trunk filtering module according to one embodiment of the present invention. In this embodiment, the trunk filtering block 430 includes load balancing logic 741 and trunk filtering logic 761. The load balancing logic 741 outputs a signal to the trunk filtering logic 761 indicating whether or not the output port corresponding to this trunk filtering module has been selected as a port on which the packet should be forwarded for purposes of load balancing. Load balancing is applied to all packets destined to a MODE 1 device and multicast packets destined to a MODE 2 device.

The trunk filtering logic 761 implements the logic discussed above with reference to the trunking rules and concepts.

FILTERING PROCESSING

Figure 7B is a flow diagram illustrating a method of trunk filtering according to one embodiment of the present invention. For each port a decision needs to be made whether to forward the current packet or filter it. The bit corresponding to this particular port in the port mask may be set to a zero to indicate not to forward the packet, for example.

At step 722, it is determined whether or not the entry retrieved from the forwarding and filtering database has directed the packet to be forwarded onto the output port. If so, one or more further tests may be performed starting at step 732. Otherwise, the filtering decision

as communicated by the filtering and forwarding database is final and the packet is filtered for this port (step 782).

Only packets that are part of a Layer 3 protocol unicast route can be forwarded over the port or trunk upon which they were received. Therefore, at step 732, a comparison is made between the input trunk number and the output trunk number. If the output trunk number is equal to the input trunk number then step 742 is executed to determine if the packet is part of a unicast route; otherwise, step 752 is executed to determine the mode of the destination device.

If, at step 742, it is determined that the packet is part of a Layer 3 protocol unicast route, then processing continues with step 752. However, if the packet is not part of a Layer 3 protocol unicast route, the packet will be filtered at step 782.

At step 752, based upon the mode of the destination device, load balancing will be applied to multicast packets (step 792) or load balancing will be applied to all packets (step 762).

LOAD BALANCING PROCESSING

Figure 7C is a flow diagram illustrating a method of load balancing according to one embodiment of the present invention.

At step 713, a hash function is performed to map packets that would otherwise be received by multiple interfaces of a multi-homed device to a single port of a trunk. Essentially, the idea is to spread the packet traffic load among the ports of a given trunk and also to prevent duplicate packets from being received at the destination. Many alternatives exist for meeting this criteria such as the exclusive or logic 644 combined with the Modulo operation 741 of Figures 6C and 7D, a random number generator, or the like.

At step 723, an output port number is determined based upon the output trunk number and the hash result. Preferably, the trunk number corresponds to the lowest number port in the trunk and the hash produces numbers in the range of zero to the number of ports in the trunk - (minus) 1. For example, if there are three trunked ports numbered 3, 4, and 5, the hash result should be a number between 0 and 2. Then, when the trunk number, 3, is added, the result is one of ports 3, 4, or 5.

At step 733, the output port number selected by steps 713 and 723, is compared to the current port number. For example, assuming a filtering block 430 were being employed for each port, the output port number of step 723 would be compared to the number of the port corresponding to the particular filtering block 430. Thus, a filtering decision is made on a port-by-port basis based upon the output port chosen by the load balancing logic. In this manner, packets destined for a MODE 1 device will only be forwarded over one of the

trunked ports coupled to the MODE 1 device. Further, multicast packets destined for a MODE 2 device will only be forwarded over one of the trunked ports coupled to the MODE 2 device.

At step 744, if it was determined that the current port corresponds to the output port selected by steps 713 and 723, then the packet will not be filtered for the current port and the packet will ultimately be forwarded over the current port. Otherwise, the packet will not be forwarded to the current port (step 753).

EXEMPLARY TRUNK FILTERING LOGIC

Figure 7D illustrates exemplary logic for implementing the trunk filtering module of Figure 7A. In the embodiment depicted, a modulus operation 791 receives the LoadBalanceHash and the OutputTrunkSize and outputs a number between zero and the OutputTrunkSize-1. An adder 793 accepts as inputs the result of the modulus operation 791 and the OutputTrunk#. After adding the OutputTrunk# (e.g., the smallest port number of the trunked ports) and the result of the modulus operation 791, the result is a number between the smallest port number and the smallest port number + OutputTrunkSize-1 inclusive (e.g., the trunked ports). The output of the adder 793 is compared to the OutputPort# by a comparator 794. At this point, a one is output if the Load balancing logic 741 has selected the output port corresponding to this trunk filtering module 430.

The next stage is the trunk filtering logic 761. In this embodiment, the trunk filtering logic 761 is implemented with a plurality of multiplexers 772-775 and unicast route logic 750. MUX 772 applies the load balancing logic results to multicast traffic. MUX 773 applies the load balancing logic results to all output ports that are coupled to MODE 1 devices (i.e., usebit is set for MODE 1 devices).

The unicast route logic 750 outputs a logic one if the packet is not a Layer 3 protocol unicast route packet and the OutputTrunk# is the same as the InputTrunk#. The unicast route logic 750 outputs a logic zero, when the packet is a Layer 3 protocol unicast route packet or the OutputTrunk# is not the same as the InputTrunk#. Thus, only packet destined for an OutputTrunk# different from the InputTrunk# or Layer 3 protocol unicast route packets can be forwarded on a given output port. It should be apparent that many other logic combinations other than an inverter 751 and an and 754 may be used.

Finally, MUX 775 filters packets that are marked as filtered by the forwarding database. Thus, no filtered packets can be "unfiltered" by the Trunk filtering block 430. That is, a PortMask value can only be changed from a one to a zero and not from a zero to a one in the FwdPortMask filter

It should be appreciated while specific field sizes, block inputs, and block outputs have been used to explain an exemplary embodiment and that the present invention is not limited to such specifics. Further, one of ordinary skill in the art will appreciate the present invention is not limited to hardwired logic. Alternatively, the logic illustrated above may be implemented in machine-executable instructions, which may be used to cause a general-purpose or special-purpose processor to produce the same results, although a relatively larger delay would likely be introduced by such an implementation.

In the foregoing specification, the invention has been described with reference to specific embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

CLAIMS

What is claimed is:

1. A network device comprising:
a plurality of ports including a first port for receiving a packet, at least the first port configured to be included in a trunk;
a memory for storing a forwarding database, the forwarding database including a plurality of entries containing therein forwarding information for a subset of addresses;

a learning circuit coupled to the first port and the memory for modifying the forwarding database to reflect an association between the first port and a first address contained within the packet, the learning circuit updating the forwarding database based upon a trunk designator corresponding to the trunk if the trunk is of a first type, the learning circuit updating the forwarding database based upon a port designator corresponding to the first port if the trunk is of a second type.

2. The network device of claim 1, further comprising a filtering circuit coupled to a second port of the plurality of ports and the memory for receiving forwarding information corresponding to a second address contained within a the packet, the filtering circuit being configured to modify the forwarding information based upon a predetermined set of trunking rules.

3. The network device of claim 2, wherein the filtering circuit comprises:
load balancing logic for selectively applying load balancing to outbound packets based upon one or more characteristics of the outbound packets and one or more characteristics of an outbound trunk; and

trunk filtering logic configured to enforce the predetermined set of trunking rules based upon one or more characteristics of the outbound trunk.

4. The network device of claim 1, wherein the forwarding database has stored therein a plurality of entries each containing an address field and a designator field, the network device further comprising a search engine coupled to the first port and to the memory for determining if the address field of an entry of the plurality of entries matches the first address.

5. The network device of claim 4, wherein the search engine includes a programmable register for each of the plurality of ports, each programmable register having stored therein a trunk designator for identifying a group of one or more ports with which the port is associated.

6. The network device of claim 5, wherein each programmable register contains a type indicator for indicating which of the trunk designator or the port designator should be used for purposes of learning.

7. The network device of claim 5, wherein each programmable register contains a trunk size indicating the number of ports that are associated with the trunk.

8. The network device of claim 1, wherein each of the plurality of ports is associated with a port designator, and wherein the trunk designator comprises the smallest port designator of all ports included in the trunk.

9. The network device of claim 6, wherein the trunk further includes one or more other ports of the plurality of ports.

10. The network device of claim 1 further including a mask generator for producing a port mask, the port mask including a mask field corresponding to each of the plurality of ports, when the mask field is in a first state the packet may be forwarded to the corresponding port, however when the mask field is in a second state the packet will not be forwarded to the corresponding port.

11. A network device comprising:

a plurality of ports including a first port for transmitting a packet to a second network device, at least the first port being included in a trunk associated with the second network device;

a memory for storing a forwarding database, the forwarding database including a plurality of entries each containing forwarding information for a particular address; and

a filtering circuit coupled to the first port and the memory for receiving forwarding information corresponding to a destination address contained within the packet, the filtering circuit being configured to modify the forwarding information based upon a predetermined set of trunking rules and one or more characteristics of the trunk.

12. The network device of claim 11, further comprising a search engine coupled to the first port and to the memory, the search engine for determining if the forwarding information of an entry of the plurality of entries corresponds to the destination address.

13. The network device of claim 11, wherein the network device further includes a learning circuit coupled to the plurality of ports and the memory, the learning circuit for modifying the forwarding database to reflect an association between a second port of the plurality of ports upon which the packet arrived and a source address contained within the packet, the learning circuit updating the forwarding database based upon a trunk designator if the second port is in a first mode, the learning circuit updating the forwarding database based upon a port designator if the second port is in a second mode.

14. The network device of claim 11, wherein the trunk further includes one or more other ports of the plurality of ports.

15. The network device of claim 11, wherein the filtering circuit further comprises load balancing logic for selectively applying load balancing to forward the packet to a single port of the plurality of ports based upon one or more characteristics of the packet and one or more characteristics of the trunk.

16. The network device of claim 15, wherein the one or more characteristics of the packet include whether the packet is a unicast packet or a multicast packet.

17. The network device of claim 15, wherein the one or more characteristics of the packet include whether the destination address is a known unicast address or an unknown unicast address.

18. The network device of claim 15, wherein the one or more characteristics of the trunk include a number of ports in the trunk.

19. The network device of claim 15, wherein the one or more characteristics of the trunk include a trunk designator associated with the trunk.

20. The network device of claim 15, wherein the one or more characteristics of the trunk include a device mode associated with the trunk.

21. A method comprising the steps of:
receiving a packet on one of a plurality of ports of a network device, the packet containing therein header information including a destination address;
searching a forwarding database for the destination address; and
if the destination address is not found in the forwarding database, then
performing load balancing to assure the packet is forwarded to only one port of a trunk if the trunk is coupled to a network device of a first type, and

forwarding the packet to all ports of the trunk if the trunk is coupled to a network device of a second type.

22. The method of claim 21, further comprising the step of if the destination address is a multicast address, then performing load balancing to assure the packet is forwarded to only one port of the trunk.

23. The method of claim 21, further comprising the steps of:
if the destination address is a known unicast address, then
performing load balancing to determine a single port of the trunk upon which to forward the packet if the trunk is coupled to a network device of the first type,
otherwise, if the trunk is coupled to a network device of the second type, forwarding the packet to the port of the plurality of ports associated with the destination address.

24. The method of claim 23, wherein network devices of the first type have the same media access control (MAC) address on all trunked ports, and wherein network devices of the second type have a different MAC address for each trunked port.

25. The method of claim 21, wherein network devices of the first type have the same media access control (MAC) address on all trunked ports, and wherein network devices of the second type have a different MAC address for each trunked port.

26. A method for use in a network device, the method comprising the steps of:
receiving a packet on a first port of a plurality of trunked ports in a trunk, each of the plurality of trunked ports associated with a different port designator and a common trunk designator;

searching a forwarding database for a source address contained in the packet; and

if the source address is not found in the forwarding database, then learning either the common trunk designator or the different port designator associated with the first port based upon a mode of the first port.

27. The method of claim 26, wherein the mode of the first port is programmable and wherein the step of learning either the common trunk designator or the different port designator further includes the steps of:

if the first port is programmed to be in a first mode, then creating a forwarding database entry containing therein a representation of the trunk designator;

otherwise, if the first port is programmed to be in a second mode, then creating a forwarding database entry containing therein a representation of the different port designator associated with the first port; and

storing the forwarding database entry in the forwarding database.

28. The method of claim 26 wherein the source address is associated with either the common trunk designator or the different port designator associated with the first port by performing the step of storing the source address in the forwarding database entry.

29. The method of claim 25 wherein the common trunk designator comprises the smallest port designator of the different port designators.

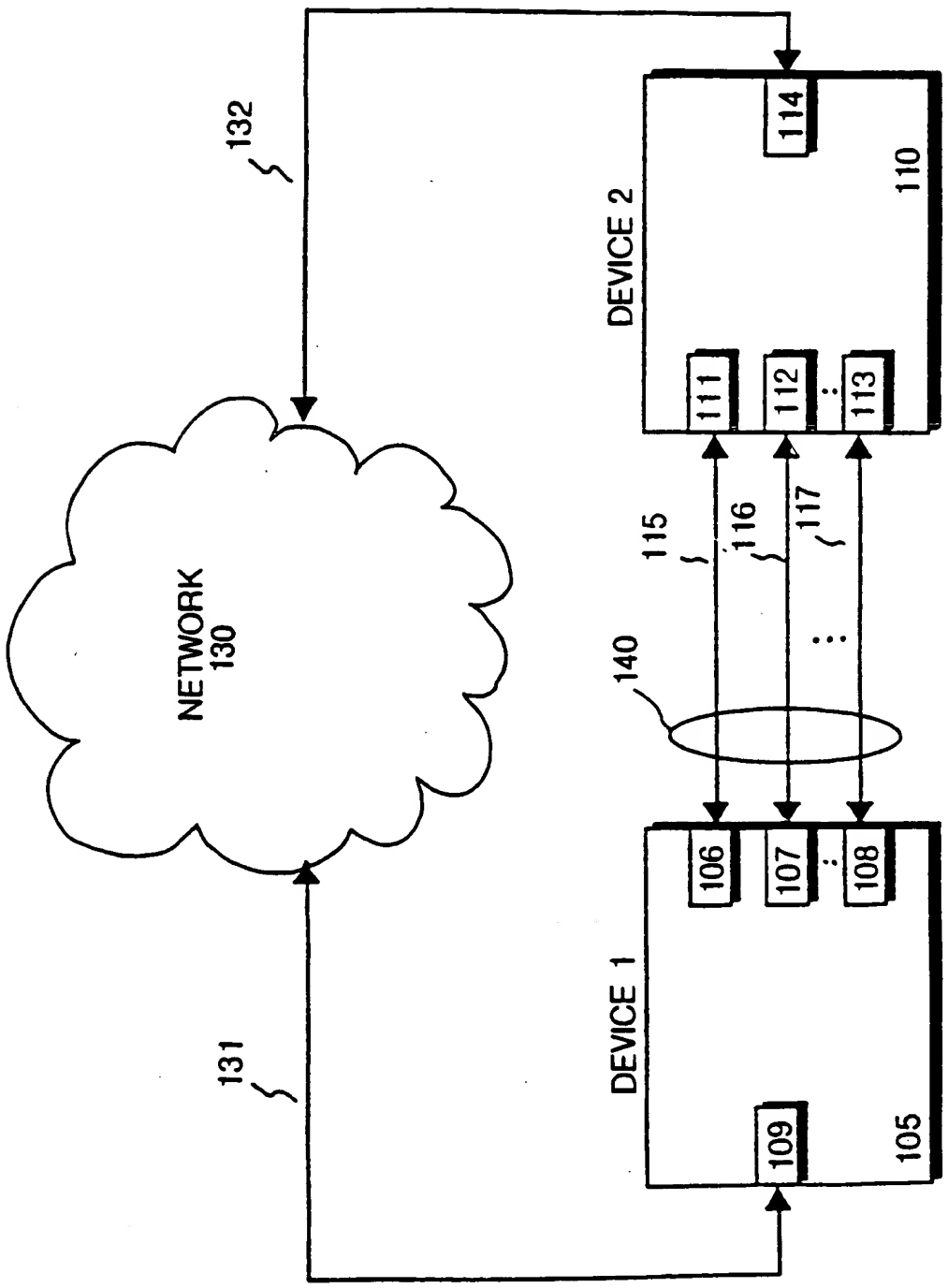

132
NETWORK
130
131
DEVICE 2
110
114
111
112
113
115
116
117
140
DEVICE 1
105
109
106
107
108

FIG. 1

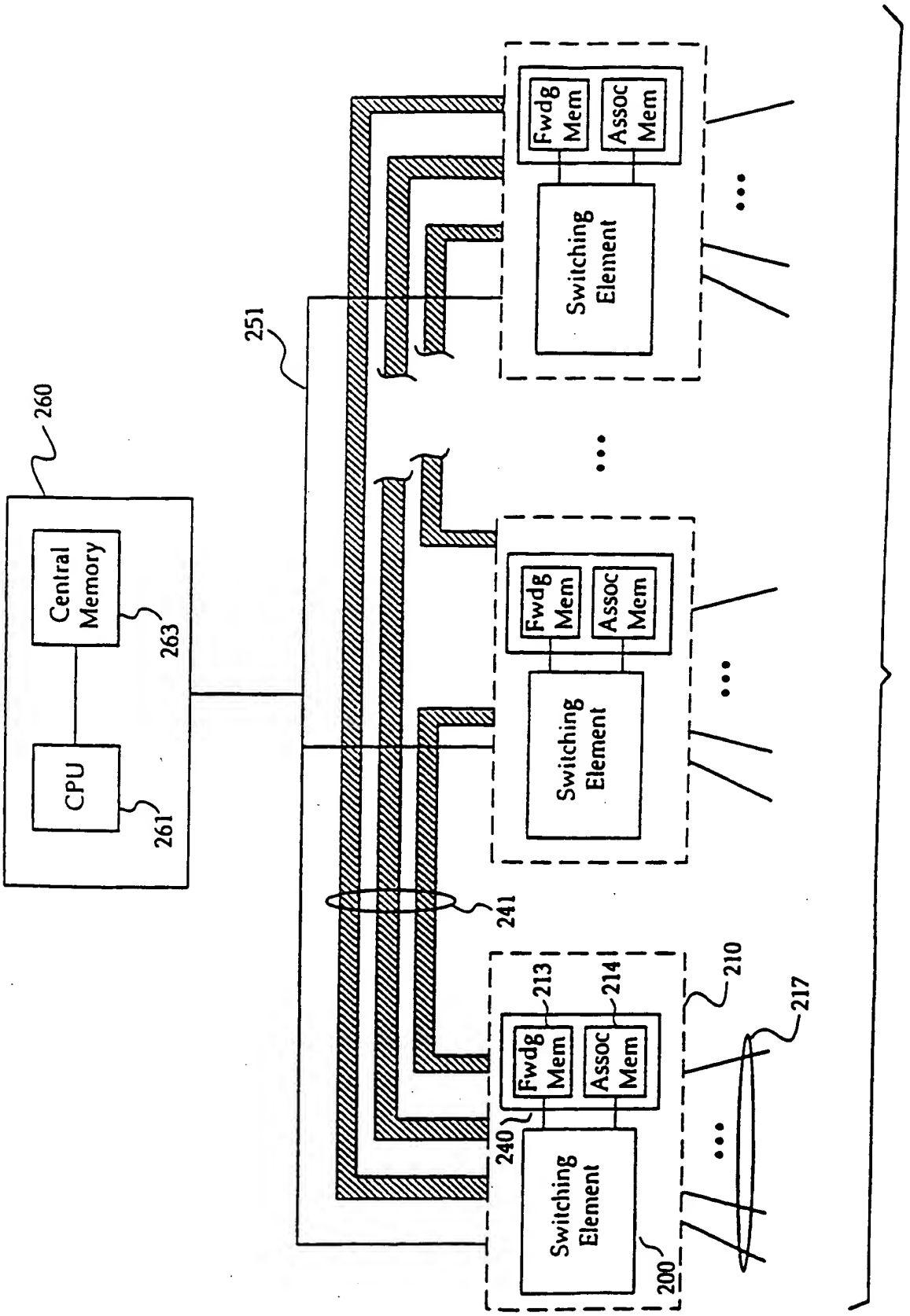

260
Central Memory
263
CPU
261
251
Fwdg Mem
Assoc Mem
Switching Element
241
213
214
240
210
217
200
201
To nodes and end-stations

FIG. 2

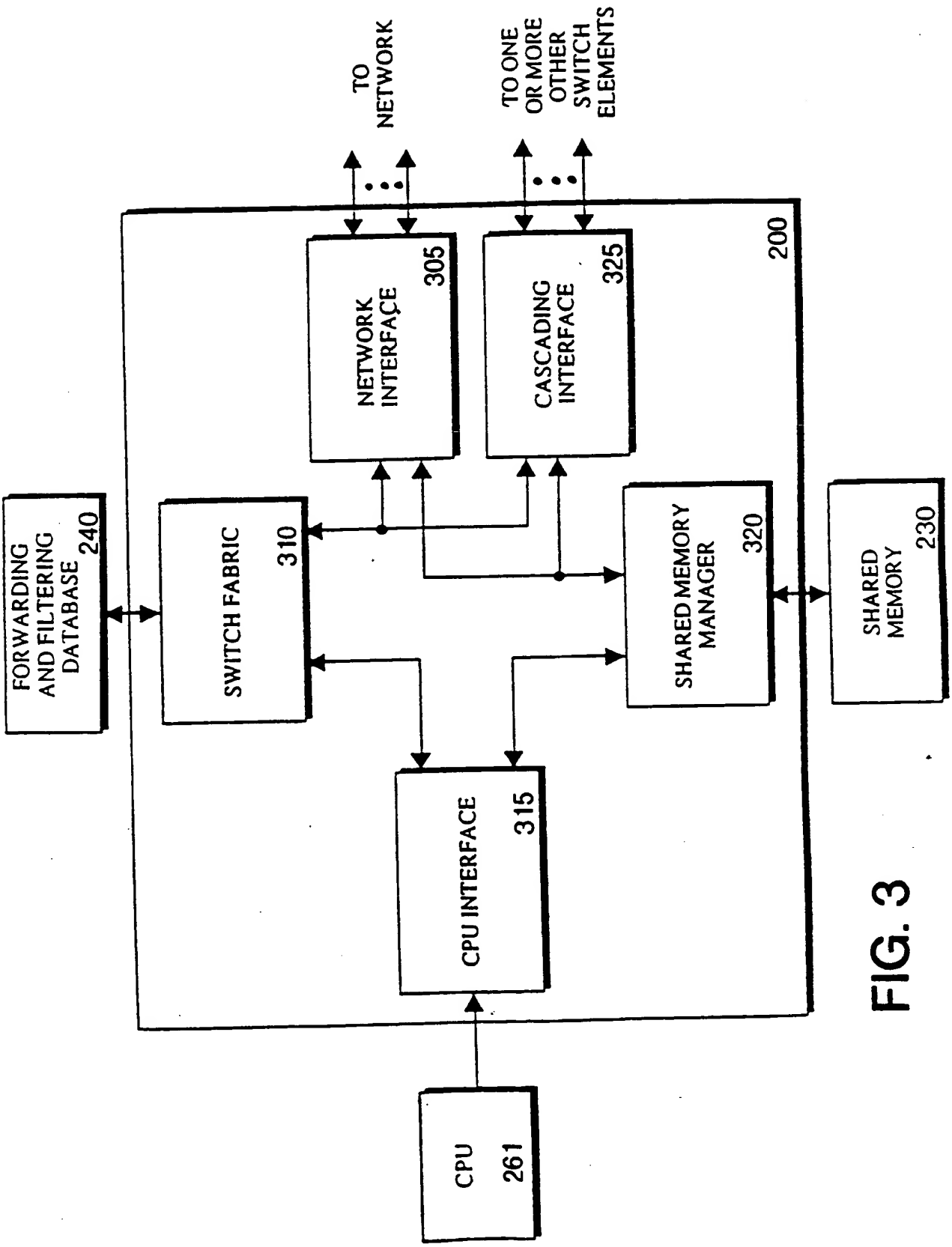

TO NETWORK
TO ONE OR MORE OTHER SWITCH ELEMENTS
200
NETWORK INTERFACE 305
CASCADING INTERFACE 325
FORWARDING AND FILTERING DATABASE 240
SWITCH FABRIC 310
SHARED MEMORY MANAGER 320
SHARED MEMORY 230
CPU INTERFACE 315
CPU 261

FIG. 3

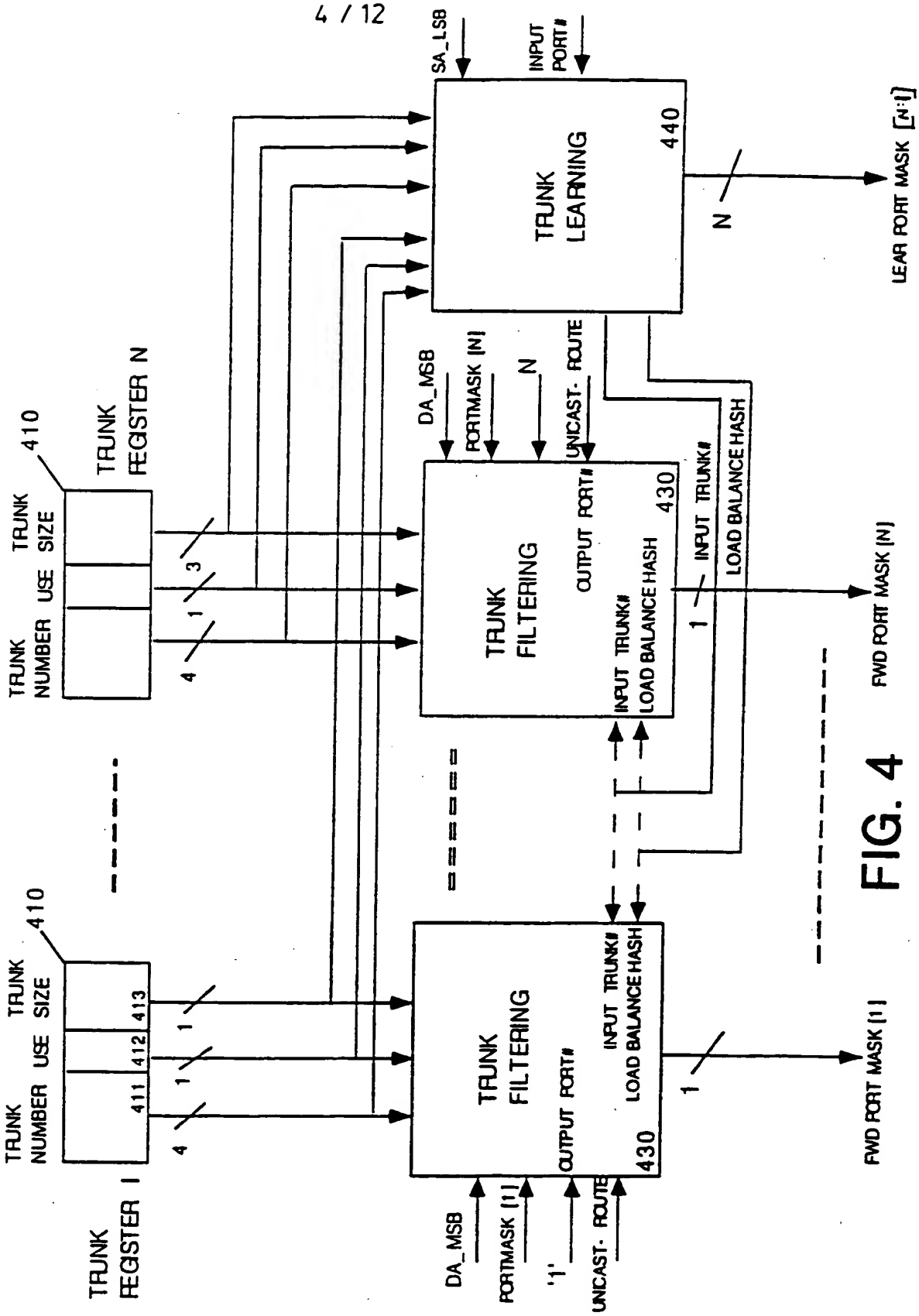


FIG. 4

SUBSTITUTE SHEET (RULE 26)

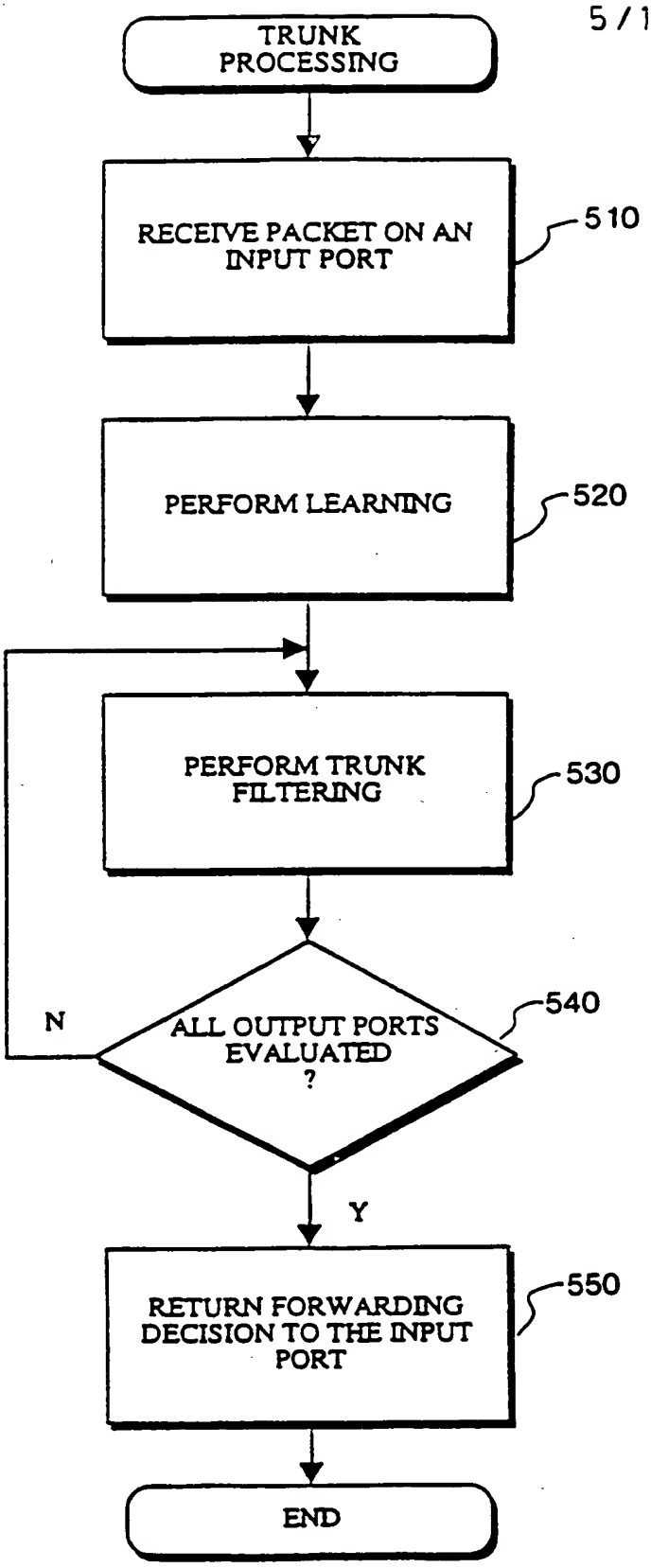

TRUNK PROCESSING
RECEIVE PACKET ON AN INPUT PORT
510
PERFORM LEARNING
520
PERFORM TRUNK FILTERING
530
ALL OUTPUT PORTS EVALUATED ?
540
N
Y
RETURN FORWARDING DECISION TO THE INPUT PORT
550
END

FIG. 5

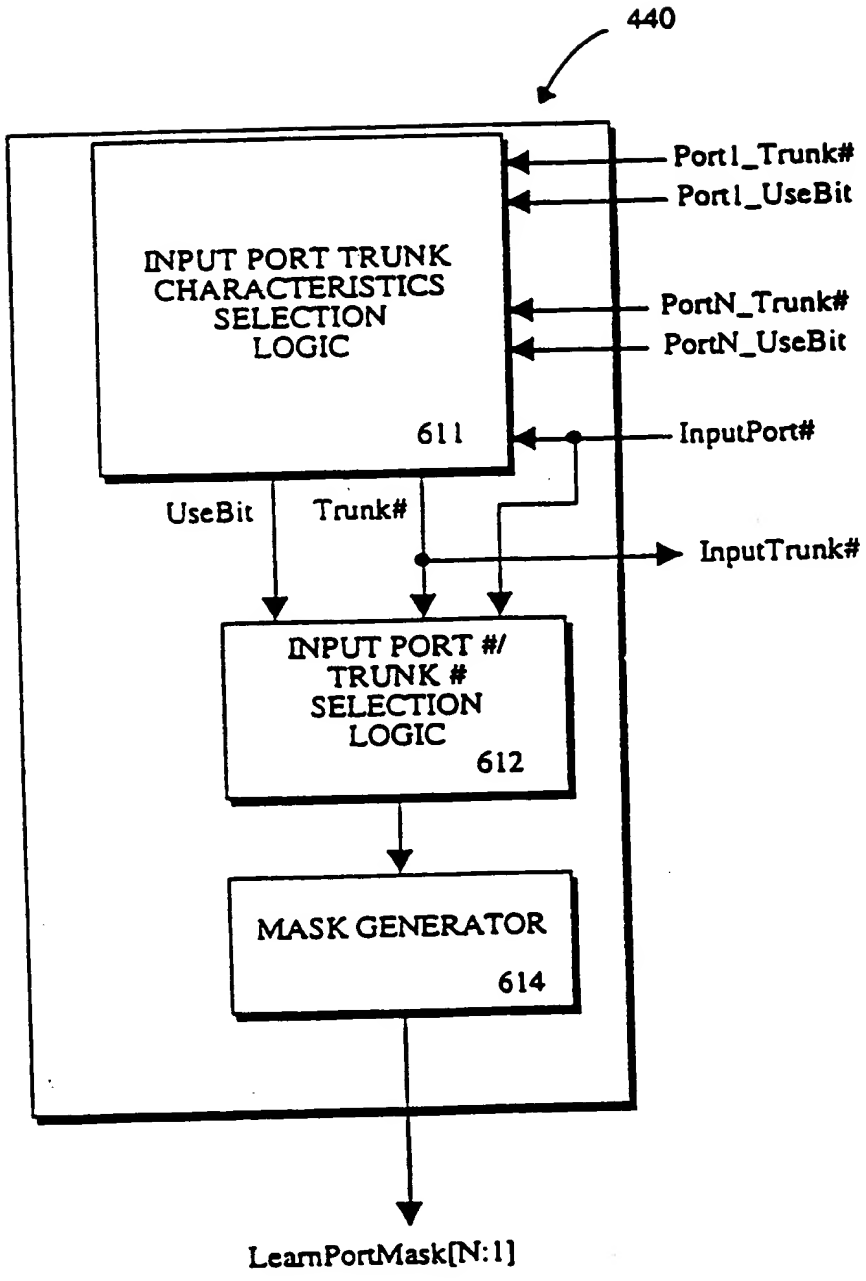

440
Port1_Trunk#
Port1_UseBit
INPUT PORT TRUNK CHARACTERISTICS SELECTION LOGIC
611
PortN_Trunk#
PortN_UseBit
InputPort#
UseBit
Trunk#
InputTrunk#
INPUT PORT #/ TRUNK # SELECTION LOGIC
612
MASK GENERATOR
614
LearnPortMask[N:1]

FIG. 6A

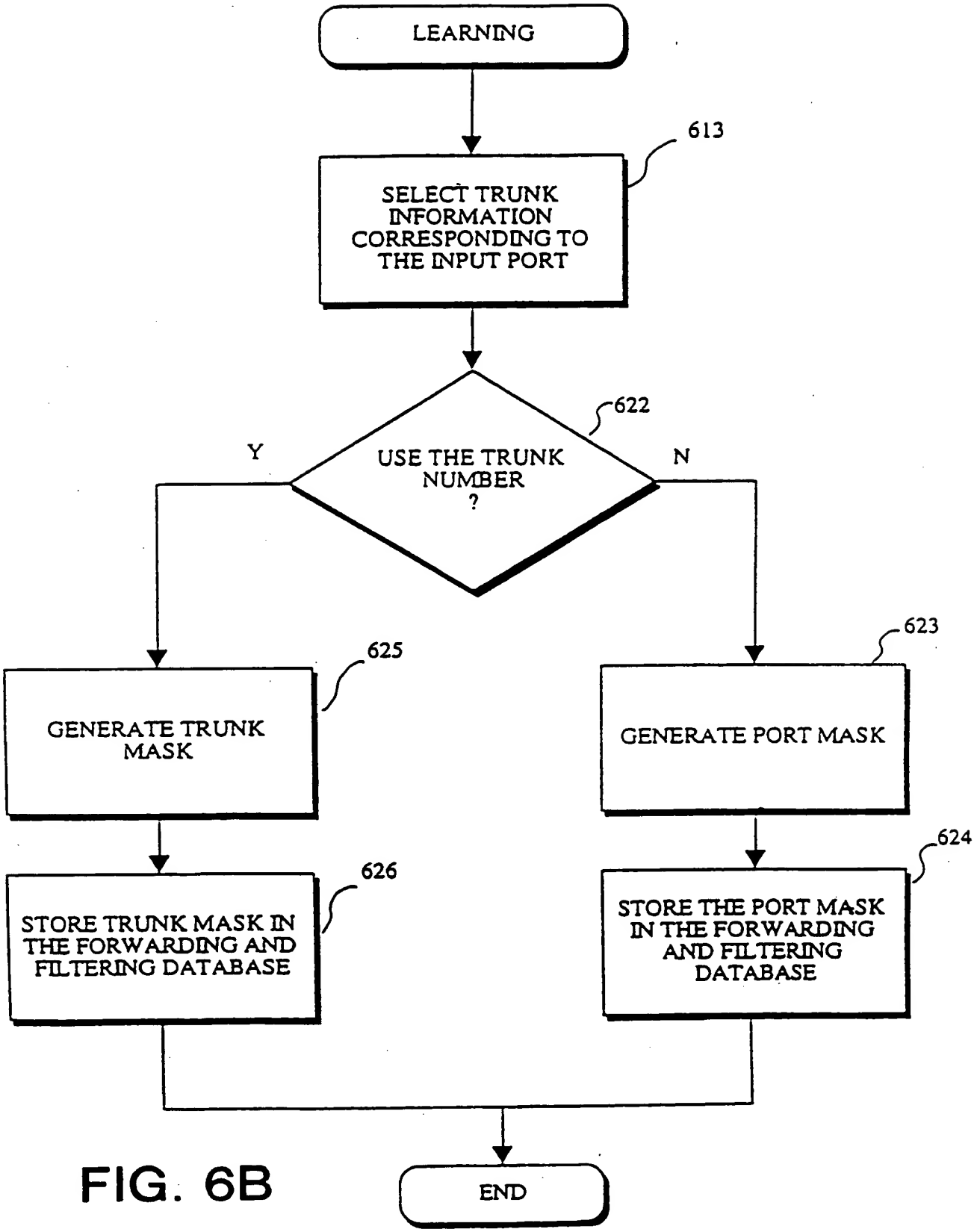

LEARNING
613
SELECT TRUNK INFORMATION CORRESPONDING TO THE INPUT PORT
622
USE THE TRUNK NUMBER ?
Y
N
625
GENERATE TRUNK MASK
623
GENERATE PORT MASK
626
STORE TRUNK MASK IN THE FORWARDING AND FILTERING DATABASE
624
STORE THE PORT MASK IN THE FORWARDING AND FILTERING DATABASE
END

FIG. 6B

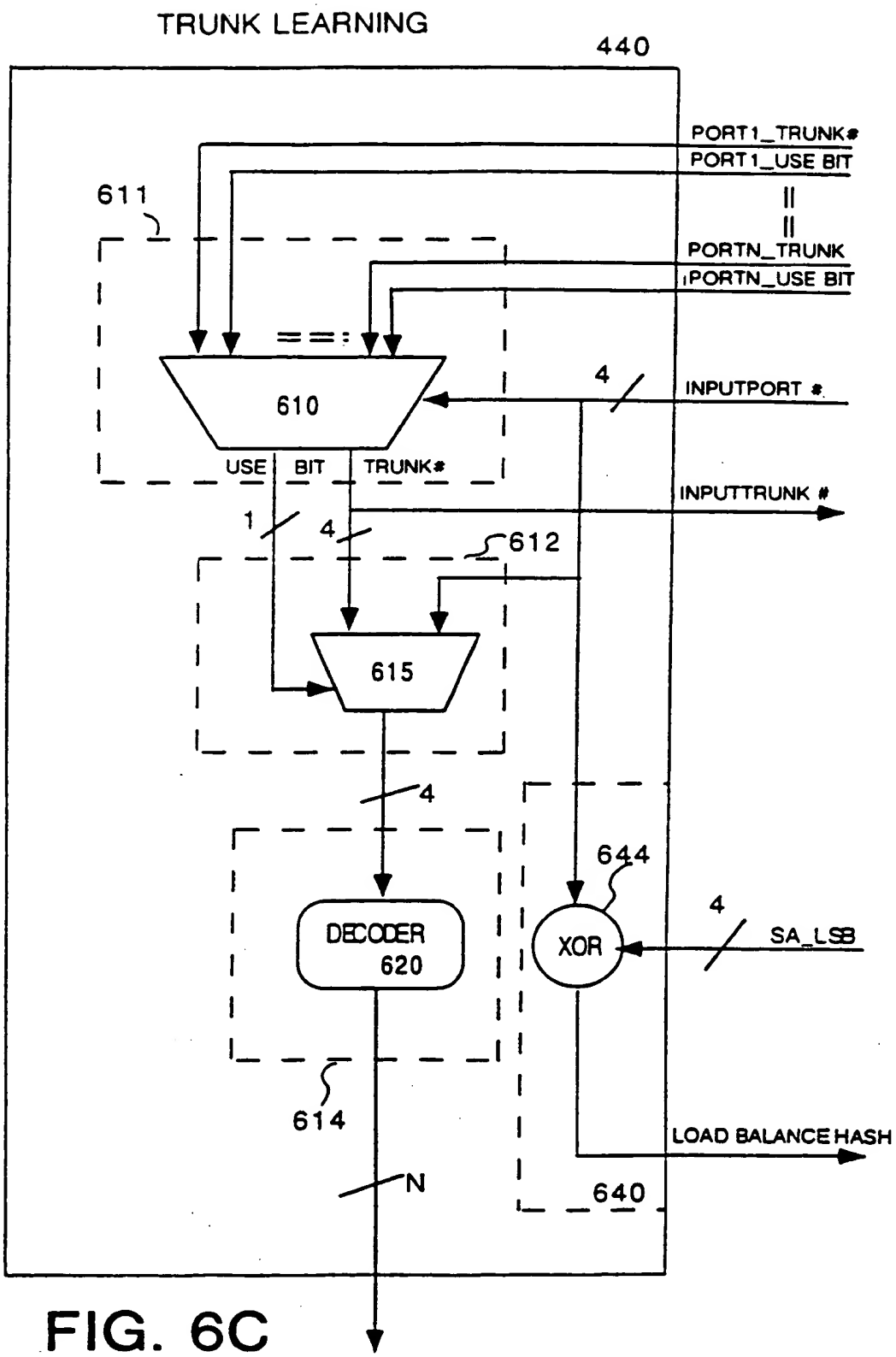

TRUNK LEARNING
440
PORT1_TRUNK#
PORT1_USE BIT
PORTN_TRUNK
PORTN_USE BIT
611
610
4
INPUTPORT #
USE
BIT
TRUNK#
INPUTTRUNK #
1
4
612
615
4
644
4
SA_LSB
DECODER
620
XOR
614
LOAD BALANCE HASH
N
640

FIG. 6C

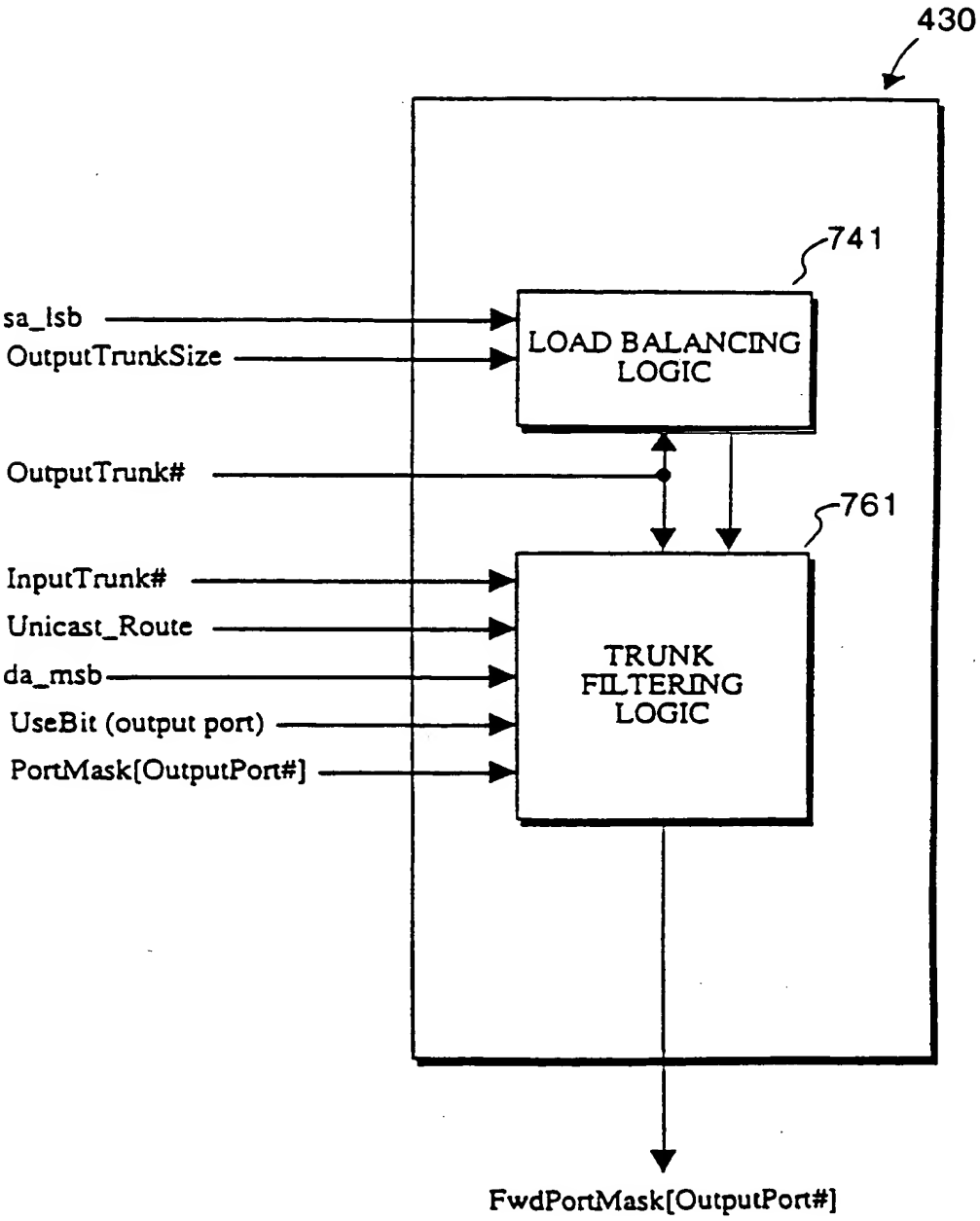

430
741
sa_lsb
OutputTrunkSize
LOAD BALANCING LOGIC
OutputTrunk#
761
InputTrunk#
Unicast_Route
da_msb
UseBit (output port)
PortMask[OutputPort#]
TRUNK FILTERING LOGIC
FwdPortMask[OutputPort#]

FIG. 7A

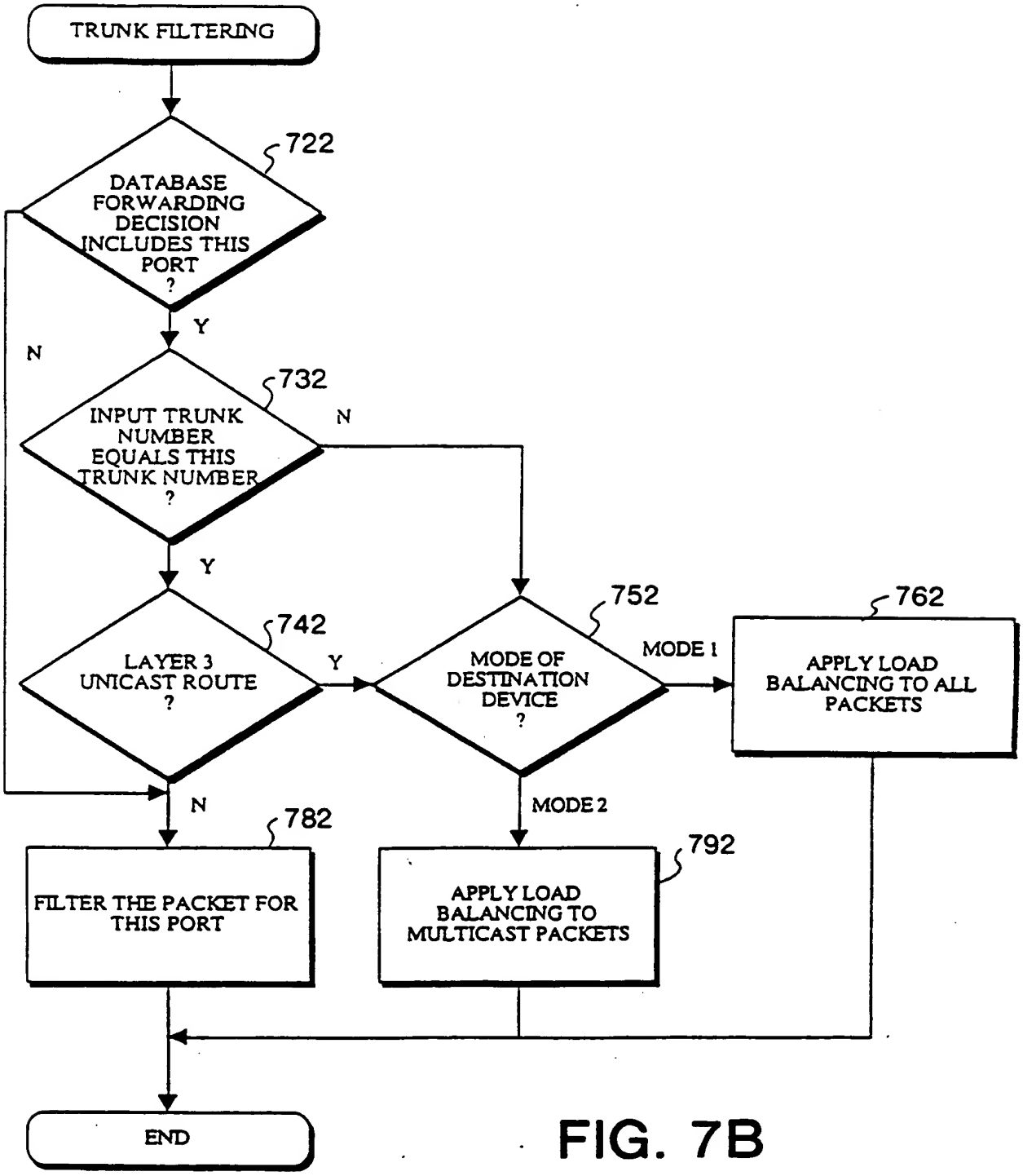

TRUNK FILTERING
722
DATABASE FORWARDING DECISION INCLUDES THIS PORT ?
Y
N
732
INPUT TRUNK NUMBER EQUALS THIS TRUNK NUMBER ?
N
Y
742
LAYER 3 UNICAST ROUTE ?
Y
752
MODE OF DESTINATION DEVICE ?
MODE 1
762
APPLY LOAD BALANCING TO ALL PACKETS
N
MODE 2
782
FILTER THE PACKET FOR THIS PORT
792
APPLY LOAD BALANCING TO MULTICAST PACKETS
END

FIG. 7B

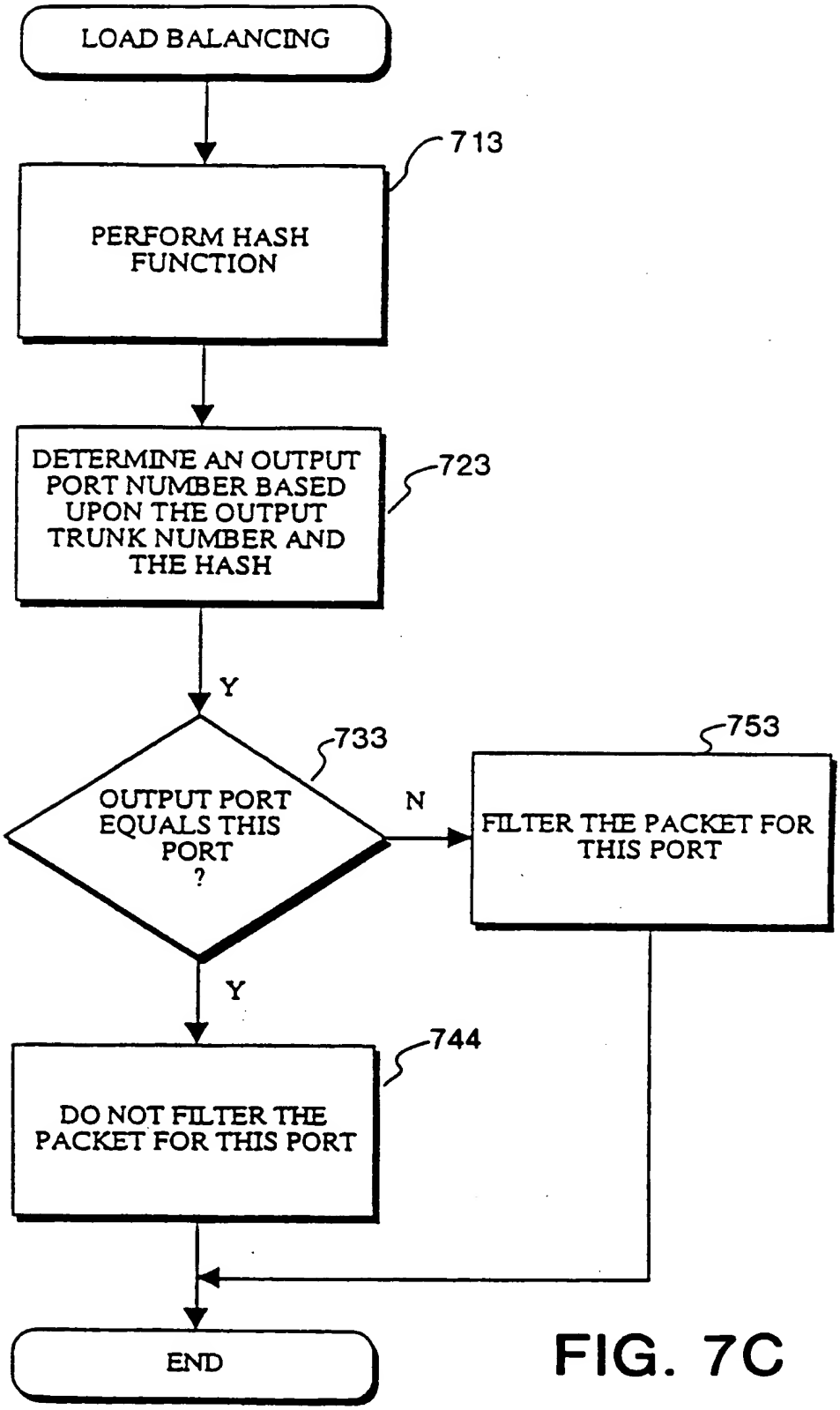

LOAD BALANCING
713
PERFORM HASH FUNCTION
DETERMINE AN OUTPUT PORT NUMBER BASED UPON THE OUTPUT TRUNK NUMBER AND THE HASH
723
Y
733
OUTPUT PORT EQUALS THIS PORT ?
N
753
FILTER THE PACKET FOR THIS PORT
Y
744
DO NOT FILTER THE PACKET FOR THIS PORT
END

FIG. 7C

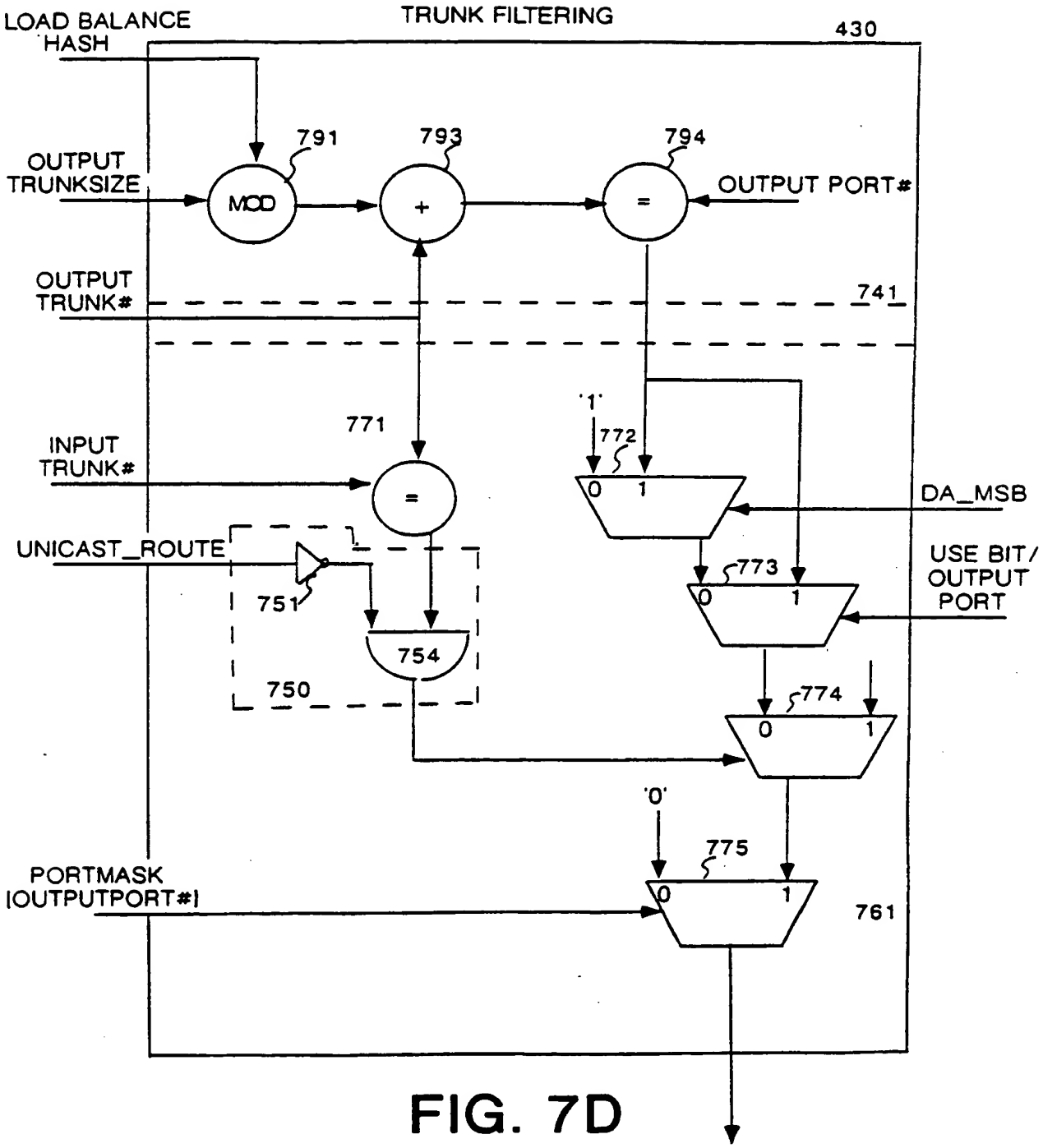

LOAD BALANCE HASH
TRUNK FILTERING
430
OUTPUT TRUNKSIZE
791
MOD
793
+
794
=
OUTPUT PORT#
OUTPUT TRUNK#
741
771
INPUT TRUNK#
=
'1'
772
0
1
DA_MSB
UNICAST_ROUTE
751
773
USE BIT/ OUTPUT PORT
754
750
774
'0'
775
PORTMASK (OUTPUTPORT#)
761

FIG. 7D

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/13368

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :H04L 12/56
US CL :370/401, 392
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 370/401, 392, 402, 466, 469, 471, 389, 390, 396, 398

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

APS
search terms: trunking support, forwarding database, multi-layer switch, bridge, router

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y ---- A	US 5,610,905 A (MURTHY et al) 11 March 1997, col. 7, line 23-col. 9, line 6.	1, 2, 4, 11, 12, 14, 21, 26 ---- 3, 5-10, 13, 15-20, 22-25, 27, 28
Y ---- A	US 5,633,865 A (SHORT) 27 May 1997, col. 2, line 52-col. 3, line 40.	1, 2, 4, 11, 12, 14, 21, 26 ---- 3, 5-10, 13, 15-20, 22-25, 27, 28
A	US 5,481,540 A (HUANG) 02 January 1996, col. 3, lines 18-58.	1-28

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 22 SEPTEMBER 1998	Date of mailing of the international search report 16 OCT 1998
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer SOON-DONG HYUN Telephone No. (703) 305-4700

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/13368

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5,570,365 A (YOSHIDA) A 29 October 1996, col. 3, lines 26-50.	1-28
A	US 5,473,607 A (HAUSMAN et al) 05 December 1995, col. 5, lines 22-56.	1-28